REMARKS

In the Office Action, the Examiner indicated that claims 1-17 are pending in the application and the Examiner rejected all claims. The rejections are respectfully traversed below.

I. Rejection of Claims 1, 2, 7, and 11-17 under 35 U.S.C. §102 and Rejection of Claims 3-6 and 8-10 under 35 U.S.C. §103(a)

At pages 3-9 of the Office Action, the Examiner rejects claims 1-17 under 35 U.S.C. §102(e) in view of Baer et al. (U.S. Patent No. 6,366,916) or 35 U.S.C. §103(a) based on a combination of Baer et al. and Rose et al. (U.S. Patent No. 5,752,244).

A. The Present Invention

The present invention is an asset locator (search engine) for locating software assets, code assets and the like that are stored in code repositories used by software designers. It provides the capability for the gathering of information about assets contained in the code repositories and the capturing of the gathered information in a database that can be used for the conducting of subsequent searches. The present invention has particular application in a software-development environment where the stored code assets may number in the millions and may be written in diverse languages such as, for example, Java, C/C++, COBOL, HTML, and/or XML.

A series of data analyzers that are specific to each type of data contained in the code repositories (e.g., a Java analyzer, a C/C++ analyzer, a COBOL analyzer, an HTML analyzer, and/or an XML analyzer) are integrated into the system so that they can be used to search the code repositories using particular attributes specific to the semantics of a particular language used to code the asset. In a preferred embodiment, the repositories are crawled automatically according to a schedule defined by the user, and the results of the crawling are stored in a database. Ordinary keyword searching can then be used with the system, either independently or combined with the attribute-specific semantic searching, to search the database.

B. Baer et al.

U.S. Patent No. 6,366,916 to Baer et al. ("Baer") is owned by IBM, the assignee of the present invention and teaches an asset manager for processing and querying assets in a database. The system comprises a Client Application layer, for manipulating and browsing assets; an Asset Manager Server layer, for providing programming interface services specific to assets types, such as storing, querying and retrieving assets representing data; and a Data Store layer, for storage of the retrieved data. Baer teaches the ability to add, update, delete, retrieve, and query Microsoft Word documents based upon properties defined in meta-data of the document, such as font size, color, typeface, paragraph starting points, etc. Baer is directed to manipulating and managing data in databases and is not directed to code repositories.

1. Rejection of Claims 1, 2, 7, and 11-17 under 35 U.S.C. §102

At pages 3-6 of the Office Action, the Examiner rejected claims 1, 2, 7, and 11-17 under 35 U.S.C. §102(e) as being anticipated by Baer et al. (U.S. Patent No. 6,366,916).

> "A claim is anticipated only if each and every element as set forth in the claim is found, either expressly or inherently described, in a single prior art reference." (Verdegaal Bros. v. Union Oil Co. of California, 814 F.2d 628, 631 (Fed. Cir. 1987) M.P.E.P. §2131.

As set forth above, the present invention is directed to locating assets from within what is typically a voluminous storage of software assets and code assets in one or multiple code repositories. It is designed to function automatically and to search for semantic "cues" that allow more focused searches to be performed on the assets, beyond the searching typically available using strictly text-based searching. Specifically, code repositories are crawled to identify code assets stored therein, and data analyzers are provided for each type of data contained in the repository. Thus, if it is known that the code repositories contain Java assets, C++ assets, COBOL assets and HTML assets, then there will be separate analyzers for each type of data. The data in the repositories is "crawled" periodically and the asset specific analyzers identify asset specific parameters

related to the stored assets and analyze them based upon these identified asset specific parameters. This enables the extraction of textual and semantic information from the stored assets, which are then stored and indexed for retrieval.

As an example, an organization might have 12 different code repositories containing assets numbering in the millions. Presume that a user wishes to locate Java programs that have utilized a hash table as the method attribute. If a prior art crawling system were used to locate any assets containing the term "hashtable" using existing text-based methods, it is highly likely that many irrelevant documents would be retrieved, including documents which contain articles related to hash tables and the like. Using the present invention, since the data crawler has performed its process not only by text terms but also by locating and indexing programs which have a method attribute (a Java-specific parameter), all Java method attributes can be searched for the term "hashtable" in a query (as would be found in a Java program utilizing a hash table in a method attribute), and the user may now retrieve only assets which have this characteristic.

Baer et al. contains no teaching of such a search capability. Baer et al. simply identifies a way of managing assets stored in a standard database. While Baer et al. recognizes that Microsoft Word files store the content of a text document and the meta-data describing font size, color, and typeface of each word in the document, Baer contains no teaching of crawling code repositories looking for assets containing asset specific semantical attributes and storing and retrieving them based upon these attributes. These elements are specifically claimed in independent claims 1 and 7 and are not taught or suggested by Baer.

Contrary to the assertion of the Examiner, column 6, lines 61-67 of Baer et al. make no mention of performing a crawl process to identify stored code assets. Likewise, column 10, lines 46-51 of Baer et al. make no mention of identifying asset specific parameters related to stored assets that have been identified via a crawl process. Further, column 13, lines 31-43 of Baer et al. contain no disclosure of analyzing stored assets based

on asset specific parameters identified based on performing a crawl process. Column 15, lines 45-56 of Baer et al. do not disclose extracting textual and semantic information from stored assets identified during a crawl process. Finally, column 15, lines 56-67 of Baer et al. do not teach or suggest storing and indexing extracted textual and semantic information for retrieval extracted from stored assets during a crawl process. Thus, each of the independent claims, and all claims depending therefrom, patentably define over Baer et al. The dependent claims include additional limitations not taught or suggested by Baer et al.

Accordingly, the Examiner is respectfully requested to reconsider and with draw the rejection of claims 1, 2, 7, and 11-17 under 35 U.S.C. §102.

2. Rejection of Claims 3-6 and 8-10 under 35 U.S.C. §103(a)

At pages 6-9 of the Office Action, the Examiner rejected claims 3-6 and 8-10 under 35 U.S.C. §103(a) as being unpatentable over Baer et al. in view of Rose et al. Baer et al., a 102(e) reference, is owned by IBM, the assignee of the present invention. Accordingly, under 35 U.S.C. §103(c), Baer et al. is not a proper reference under 35 U.S.C. §103 and is not available for use in making an obviousness rejection. Accordingly, the Examiner is requested to reconsider and withdraw the rejection of claims 3-6 and 8-10 under 35 U.S.C. §103.

II. The Abstract

The Examiner indicated on page 2 of the Office Action that the Abstract of the disclosure was objected to. Applicants submit herewith a replacement Abstract which contains only one paragraph as required by the Examiner.

III. The Drawings

The Examiner indicated on page 2 of the Office Action that Figure 2 should contain "reference character 42" as mentioned in the description. Applicants enclose a proposed drawing correction for the Examiner's approval.

IV. Claim Amendments

Applicants have amended claims 15-17 to correct an obvious typographical error.

V. Conclusion

The rejection of claims 1-17 have been traversed. Accordingly, reconsideration of the present application, and withdrawal of the rejections on the grounds of 35 U.S.C. §102 and §103 is respectfully requested.

Attached hereto is a marked-up version of the changes made to the claims by the current amendment. The attached page is captioned "Version with Markings to Show Changes Made".

The Commissioner is hereby authorized to charge any fees associated with this communication to Deposit Account No. 09-0461. This Reply is enclosed in duplicate.

Respectfully submitted,

August 12, 2002
Date

Mark D. Simpson, Esq.
Registration No. 32,942
SYNNESTVEDT & LECHNER LLP
Suite 2600 Aramark Tower
1101 Market Street
Philadelphia, PA 19107
Telephone: (215) 923-4466
Facsimile: (215) 923-2189

VERSION WITH MARKINGS TO SHOW CHANGES MADE

Claims 15-17 have been amended as follows.

15. The system [method] as described in claim 11, wherein said stored assets comprise code assets and wherein said asset-specific parameters comprise languages in which each code asset is written.

16. The system [method] as described in claim 15, wherein said analysis step is performed using language-specific analyzers corresponding to the languages of said code assets.

17. The system [method] as described in claim 16, wherein said language-specific analyzers analyze said stored assets based on predetermined parameters specific to the language to which they correspond.

The Abstract has been amended as follows.

A method and system for locating assets is disclosed that provides the capability for the gathering of information about assets contained in data repositories and the capturing of the gathered information in a database which can be used for the conducting of subsequent searches. [The information can be captured from a single data repository or from plural data repositories, e.g., all data repositories maintained by a particular enterprise in disparate locations, with the captured information being consolidated into a single database for access by multiple users. The present invention has particular application in a software-development environment where the software assets may number in the millions and may be stored in plural languages such as, for example, Java, C/C++, COBOL, HTML, and/or XML]. By integrating into the system a series of data analyzers that are specific to each type of data contained in the repositories (e.g., a Java analyzer, a C/C++ analyzer, a COBOL analyzer, an HTML analyzer, and/or an XML analyzer) the system can be used to search the repositories using conventional keyword-searching techniques, as well as by searching for particular attributes specific to a particular language, or by combining keyword-searching with attribute-specific searching.

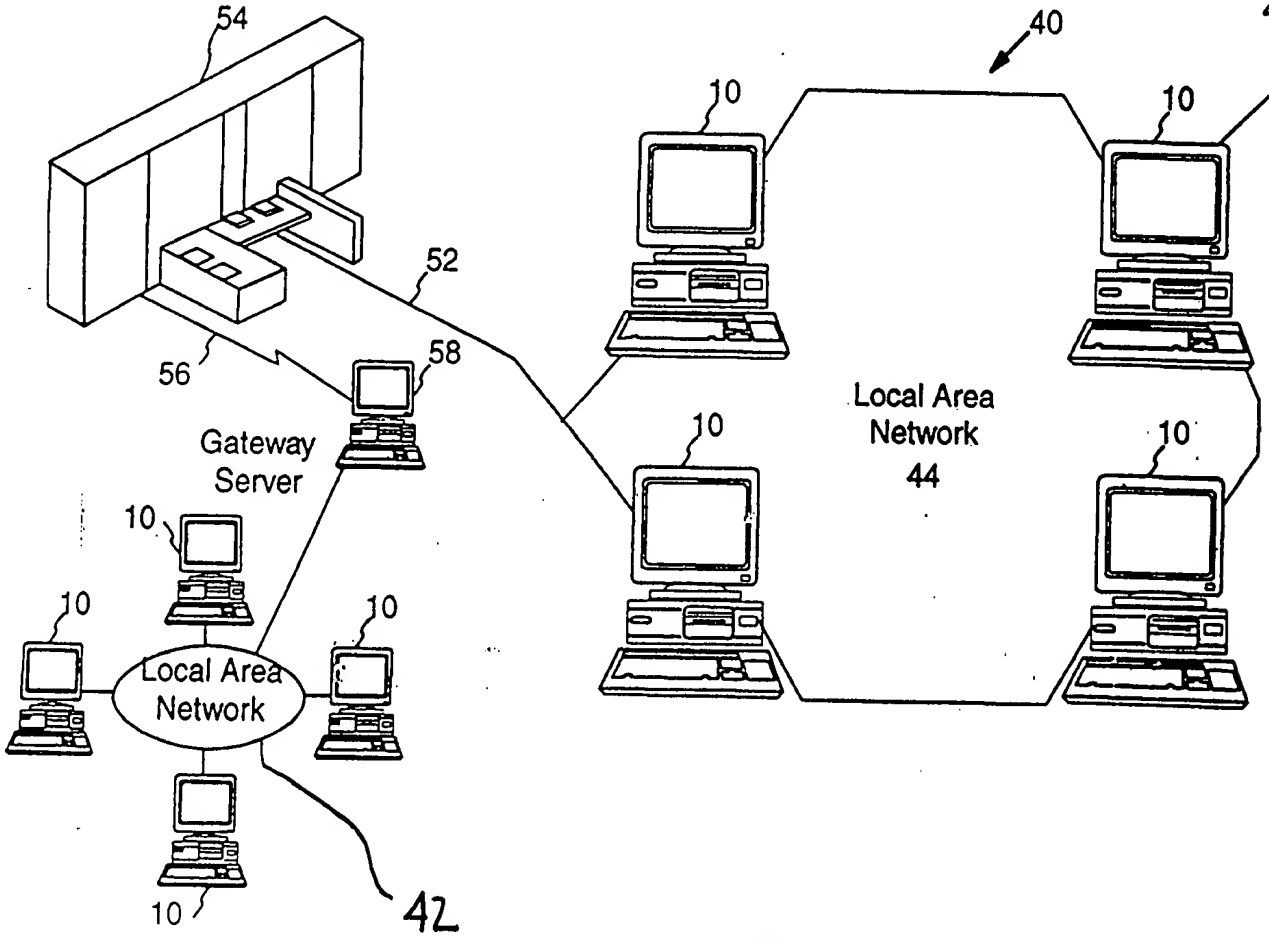
54
40
48
46
10
10
52
50
56
58
Local Area
Network
44
Gateway
Server
10
10
10
10
Local Area
Network
10
42

FIG. 2